Resümee: Konzepte der Vorlesung

- Prinzipielle Phänomene und Begriffe herausarbeiten
  - Kausalität, Konsistenz, verteilte Berechnung, safety und liveness,...
- Geeignete Modelle und Abstraktionen entwickeln
  - z.B. Zeitdiagramme, Atommodell, Zustandsgitter, Gummibandtransform.
- Problemlösungs-, Analyse- und Verifikationstechniken
  - z.B. Beweis über Invarianten
- Techniken, Einsichten, Zusammenhänge,...
  - Komplexitätsanalyse
  - Transformationen zwischen Problemklassen
  - Problemverständnis von einem höheren Standpunkt

Resümee (1)

- Verteilte Systeme
- Kooperation durch Kommunikation
  - keine globale Sicht
  - keine gemeinsame Zeit
  - parallel
  - nicht-deterministisch
  - unbestimmte Nachrichtenlaufzeit
- Typische Probleme verteilter Systeme / Algorithmen:
  - Beobachtungsproblem (keine Gleichzeitigkeit)
  - Schnappschussproblem (wieviel Geld ist in Umlauf?)
  - Terminierungserkennungsproblem

Resümee (2)

- Typische Probleme verteilter Systeme / Algorithmen:
  - Deadlockproblem (Phantomdeadlock?)
  - Kausalitätsproblem (indirekte Wirkung vor Ursache)
- Problem globaler Prädikate ("relativistischer Effekt")
  - es gibt i.a. mehrere "gleichberechtigte" Beobachter
  - diese stimmen i.a. bzgl. der Gültigkeit des Prädikates nicht überein!
  - gibt es beobachterinvariante Prädikate?
- Verteilter Euklidischer Algorithmus
  - als erstes Beispiel für einen verteilten Algorithmus
  - reaktives Verhalten ("nachrichtengesteuert")
  - Korrektheit der Idee / des konkreten Algorithmus? (Invarianten...)

Resümee (3)

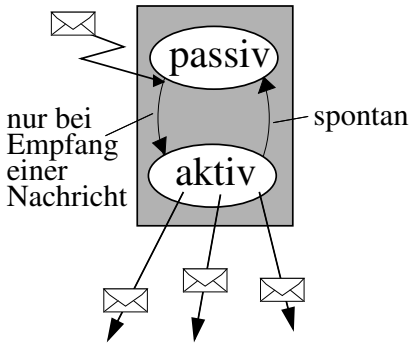
- Zahlenrätsel
  - Parallele Constraint-Propagation
  - Abwechselnd mit Backtracking-Schritt

allgemeines Schema ("verteilte Approximation")

- Konzeptuelle Hilfsmittel
  - Zeitdiagramme
  - Atomare Aktionen
- Problem der verteilten Terminierung
  - Geeignete Definition?
  - Verfahren zur Feststellung?
- Flooding-Algorithmus
  - Nachrichtenzahl
  - Problem der Terminierungserkennung
  - Formalere Fassung in Pseudo-Code
- Echo-Algorithmus (Variante von Flooding)
  - Nachrichtenzahl 2e
  - Explorer- / Echo-Welle
  - Spannbaum
  - Zwei "disjunkte" Wellen (rot; grün)

Resümee (4)

- Echo-Algorithmus
  - Überlagerung; upcall / downcall
  - Formalere Fassung in Pseudo-Code
  - Verbesserung durch Mitführen von Knotenidentitäten?
- Zeitkomplexität
  - Einheitszeitkomplexität
  - Variable Zeitkomplexität
- Paradigma der verteilten Approximation
  - Verallgemeinerung verschiedener ähnlicher Algorithmen
- Verteilte Terminierung
  - Problemdefinition
- Atommodell
  - Vereinfacht die Betrachtung des Wesentlichen
  - Terminierungskriterium: "keine Nachricht unterwegs"



- Schiefes Bild beim Beobachten verteilter Berechnungen

  --> Pauschales Zählen von Nachrichten genügt nicht zur Erkennung der verteilten Berechnung

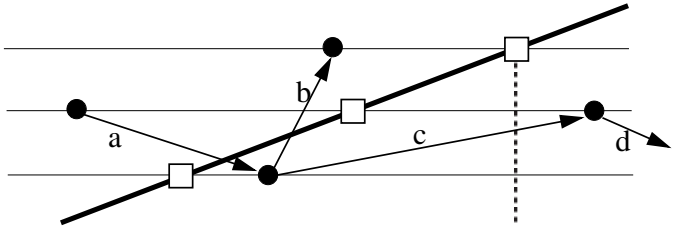
- Besprechung Übungen (1): verteilte ggT-Berechnung
  - Varianten des Algorithmus (z.B. andere Topologien)
  - Verifikationsidee (Invarianten etc.)

Resümee (5)

- Schiefes Bild beim Beobachten verteilter Berechnungen

  --> Pauschales Zählen von Nachrichten genügt nicht zur Erkennung der verteilten Berechnung

  --> Suche nach den eigentlichen Ursachen für Fehlschlag des Zählkriteriums ("Kompensation" der Zähler")



- Lösungsansätze zur Terminierungserkennung
  - Durch Vermeidung der "Ursachen" für das schiefe Bild
- Terminierungserkennung: *Eindeutige Nachrichtennamen*
- Terminierungserkennung: *Kanalzählerkriterium*
  - Widerspruchsbeweis (es gibt kein frühestes Ereignis nach dem Schnitt)
- Terminierungserkennung: *Doppelzählverfahren*
  - informeller Beweis (Aussage über gedachten senkrechten Schnitt zwischen den beiden Wellen)
  - formalerer Beweis in den Folienkopien

- Routing, Broadcast beim Hypercube und Butterfly

Resümee (6)

- Terminierungserkennung: *Doppelzählverfahren*
  - Eigenschaften

- Safety- und Liveness-Eigenschaften verteilter Algorithmen

- Kontrolltopologien zur Realisierung von Schnitten
  - Ring
  - Spannbaum
  - Echo-Algorithmus als zugrundeliegendes Basisverfahren
    (Hin- und Rückwelle für die beiden Schnitte des Doppelzählverfahrens!)

- Terminierungserkennung: *Zeitzonenverfahren*
  - Prinzip: *Erkenne* "Nachricht aus der Zukunft"
  - binäre "schwarz/weiss"-Zeit genügt

Resümee (7)

- Terminierungserkennung: *Zeitzonenverfahren*
  - binäre "schwarz/weiss"-Zeit genügt
  - Qualitätseigenschften im Vergleich zu anderen Verfahren

- Diskussion: Terminierungserkennung durch *Einfrieren*?

- Terminierungserkennung: Vermeiden inkonsistenter Schnitte durch geeignetes *Vorziehen der Schnittlinie*

- Synchrones / asynchrones Senden
  - synchron: senkrechte Nachrichtenpfeile sind gerechtfertigt
  - nicht alles geht synchron (z.B. Überholen von Nachrichten)

- Besprechung Übung 1: Formalisierung von Zeitdiagrammen und "kausal abhängig"

Resümee (8)

- Besprechung Teile von Übung 1
  - kausaltreue Beobachtungen als lineare Erweiterungen ("Einbettung") der halbgeordneten Kausalitätsrelation

- Charakterisierung synchroner Kommunikation
  - alle Nachrichtenpfeile können senkrecht gezeichnet werden; Kommunikationskanäle sind immer leer
  - es gibt eine lineare Erweiterung der Kausalitätsrelation, so dass ein Empfangsereignis immer direkt nach seinem Sendeereignis kommt
  - Senden und Empfangen bilden "atomare Einheit"
  - Nachrichten-Scheduling-Relation (m < n gdw. send(m) < receive(n)) ist zyklenfrei
  - Zyklenfreiheit der "synchronen Kausalitätsrelation <<" ("common past" / " common future"); dadurch Identifizierung von send und receive

zusammengehörige send/receive-Ereignisse sind "in gewissem Sinne" atomar

- Fragen...
  - sind die Charakterisierungen alle äquivalent?
  - kann man nun Nachrichtenlaufzeiten immer vernachlässigen?
  - funktioniert ein Algorithmus, der unter der Voraussetzung synchroner Kommunikation gemacht wurde, auch bei asynchroner Kommunikation?
  - und umgekehrt?
  - Terminierungserkennung bei synchroner Kommunikation? (das Atommodell ist dann offenbar nicht mehr adäquat, oder?)

Resümee (9)

- Def. verteilte Terminierung bei synchroner Kommunikation

X_p: {$state_p$ = *aktiv*}
  $state_q$:= *aktiv* (* "atomares" aktivieren *)

I_p: $state_p$:= *passiv*

- Verhaltensmodelle verteilter Anwendungen
  - Transaktionsmodell
  - Atommodell
  - Synchronmodell

  } gegenseitige Simulation bzw. Transformation der Modelle

- Modelle in der Informatik
  - nicht nur zum Erkenntnisgewinn, zur Simulation etc., sondern auch Implementierung von "ausgedachten, idealisierten Wirklichkeiten"

- Terminierungserkennung bei synchroner Kommunikation
  - z.B. Erkennen einer senkrecht von oben nach unten laufenden Nachricht, die einen schrägen Schnitt ("Welle") überquert

- Algorithmus von Dijkstra et al. ("DFG")
  - schwarz / weiss-Färbung; Token auf einem Kontrollring
  - Beschreibung durch Menge von Verhaltensregeln
  - Überlegungen zu Korrektheit, Varianten, Nachrichtenkomplexität

Resümee (10)

- Wechselseitiger Ausschluss
  - safety
  - liveness
  - fairness
  - nachrichtenbasierte Lösung ("requests" etc.): zentraler Manager
  - andere Lösung: Token-Ring
- Maekawa's $O(\sqrt{n})$-Algorithmus
  - Prinzip: Gitteranordnung

Resümee (11 / 12)

- Logische Zeit
  - Zeitstempel von Ereignissen
  - Uhrenbedingung. kausale Unabhängigkeit
  - logische Uhren von Lamport
    - Definition
    - Realisierung
    - injektive Abbildung, eindeutige Zeitpunkte
- Wechselseitiger Ausschluss
  - verteilte Warteschlangen von Lamport
  - Lösung Ricart / Agrawala 1981
  - Lösung Ricart / Agrawala 1983
- Wechselseitiger Ausschluss: *token-basierte* Algorithmen
  - Spannbaum / Baum: Umdrehen durchlaufener Kanten ("path reversal")
  - "Lift-Algorithmus" --> O(log n) bei "guten" Bäumen
  - Verallgemeinerung auf beliebige (gerichtete azykl.) Graphen
  - Invarianten: Zyklenfreiheit; alle Pfade führen zum Tokenbesitzer
  - Variante: Nachbarn informieren, dass Token "jetzt" hier ist (Quittungen?)
  - Request holt Token stets ein
  - spezielle Topologien (Ring; Stern; lineare Kette)
  - Nachrichtenkomplexität bei starker Last (≈4)

Resümee (13/14)

- Vergleich von Algorithmen für den wechselseitigen Ausschluss (quantitative und qualitative Kriterien)

- Parallele Berechnungsschemata
  - Bsp.: Integration mittels Trapezmethode
  - Lastausgleich durch Migration von Arbeitseinheiten
  - Gesamtlast = 0 <==> Terminierung
- Terminierungserkennung mit der Kreditmethode
  (Halbieren von Tickets; Einsammeln von "Krümeln")
- Safety: "Gesamtkredit" ist invariant
- Realisierung in verschiedenen Varianten möglich:
  - geeignete Darstellung der Krümel (negativer Zweierlogarithmus)
  - geeignete Realisierung des Einsammelns (Liveness!)
  - geeignete Verwaltung der Krümel bei den Prozessen
  - geeignete Informationsverwaltung im Urprozess
- Nachrichtenkomplexität: Worst-case-optimal
- Variante: direktes Nachlaufen
  - Analogie zum Echo-Algorithmus!

- Besprechung von Teilen von Übung 3
  - falscher Terminierungserkennungsalgorithmus
  - es genügt nicht, nur über den Zustand seiner Nachbarn informiert zu sein

Resümee (15)

- Besprechung von Teilen von Übung 3
  - falscher Terminierungserkennungsalgorithmus
  - es genügt nicht, nur über den Zustand seiner Nachbarn informiert zu sein
- *Election-Problem*: Symmetriebrechung
  - Auswahl genau eines Prozesses aus mehreren (bis auf die eindeutige Identität) gleichartigen
- Election-Algorithmus mit dem Message-extinction-Prinzip
  - verteiltes Approximationsschema
  - funktioniert auf allgemeinen (zusammenhängenden) Graphen
  - aber: Problem der Terminierungserkennung
- Election-Algorithmus auf (unidirektionalem) Ring
  - nur grösste Identität schafft Ringumlauf --> ist damit "gewählt"
  - Bully-Algorithmus, oder besser:
  - message-extinction (beim Ring kein Terminierungserkennungsproblem!)

Resümee (16)

- Chang/Roberts-Algorithmus auf unidirektionalem Ring
  - Worst-case-Nachrichtenkomplexität $O(n^2)$
- Chang/Roberts-Algo.: Mittlere Nachrichtenkomplexität?
  - Wahrscheinlichkeit, genau i Positionen weit zu kommen
  - Erwartungswert für die Länge der Nachrichtenkette = H_n
  - mittlere Nachrichtenkomplexität = nH_n (= ca. n ln n)
- Besprechung von Teilen von Übung 4
  - Wartezeit bis zum ersten Rekord
  - Simulation eines "unendlichen" Erwartungswertes??

Resümee (17)

- Qualität des Chang/Roberts-Algorithmus
  - worst-case; average-case
- Bidirektionale Varianten des Chang/Roberts-Algorithmus
  - probabilistisch / deterministisch (z.B. bei Kenntnis der Nachbaridentitäten)
  - mittlere Nachrichtenkomplexität

Resümee (18)

- Algorithmus von Hirschberg und Sinclair (bidir. Ring)
  - sukzessive grössere Gebiete erobern
  - worst-case Nachrichtenkomplexität < 8 n $\log_2$ n
- Petersons Election-Algorithmus (bidir. Ring)
  - solange sukzessive Identität in beide Richtungen senden, bis man von einem grösseren Nachbarn erfährt
  - worst-case Nachrichtenkomplexität < 2 n $\log_2$ n
  - mittlere Nachrichtenkomplexität ca. 2 n $\log_3$ n
  - Simulation ("kostenneutral"!) auf einem unidirektionalen Ring
  - Variante mit abwechselnden Richtungen
  - worst-case Nachrichtenkomplexität (ca. 1.44 n $\log_2$n+ c) mittels Fibonacci-Folge abgeschätzt

Resümee (19)

- Election auf Bäumen
  - Explosionswellen vereinigen sich
  - Explosionswelle wird an den Blättern reflektiert
  - Kontraktionsphase endet in zwei Zentrumsknoten
  - Nachrichtenkomplexität O(n)
- Echo-Election auf allgemeinen Graphen
  - Idee wie Chang/Roberts, aber Echo-Algorithmus statt Ringumlauf
- Nachrichtenkomplexität des Election-Problems
  - mindestens e Nachrichten

Resümee (20)

- Verteilte Spannbaumkonstruktion
  - Zusammenhang zum Election-Problem ("gleich schwierig")
- Anonyme Netze
  - De-Anonymisierung
- Election in anonymen Netzen
  - kein stets terminierender (deterministischer) Algorithmus möglich
- Probabilistische Algorithmen
  - Las Vegas (terminiert nicht immer, Ergebnis ist aber korrekt)
  - Monte Carlo (terminiert, aber ggf. mit falschem Ergebnis)
- Probabilistische Election-Algorithmen
  - Itai / Rodeh: Chang/Roberts-Verfahren mit Zufallsidentität
  - Implementierung reellwertiger Zufallszahlen zwischen 0 und 1?

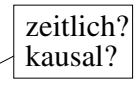
- Kausaltreue Beobachtungen
  - Beispiel: Aussterben aller Exemplare eines "Typs" (--> Terminierung)
  - Analogie: konsistente Referenzzähler (--> Garbage-Collection!)
  - Lösungen: Synchrone Kommunikation oder getrennte FIFO-Puffer pro Prozess für das Empfangen und Senden von Nachrichten

---

- Garbage-Collection
  - Objekte und Zeiger; Wurzelobjekte
  - nicht mehr von der Wurzel erreichbar --> Garbage
  - rekursives Freigeben (Zyklen bleiben übrig!)
  - *Mutator* (new, copy, delete: Manipulation von Zeigern)
  - *Collector* soll Garbage-Objekte identifizieren

Resümee (21)

- Garbage-Collection
  - rekursives Freigeben (Zyklen bleiben übrig!)
  - Paradigmen: "stop the world" / on the fly (= "parallel")
  - "Mark and sweep"-Verfahren
  - bei paralleler Variante: Problem mit "behind the back copy"
    ==> Mutator / Collector müssen sich koordinieren!
    (sonst bekäme der Collector ggf. ein "schiefes Bild")
- Verteiltes Garbage-Collection (= GC in verteilten Systemen)
  - Referenzen u.U. "in transit"
  - copy nicht mehr atomar ("send/receive copy")
  - increment / decrement per Nachricht (z.B. an den Ort des Referenzzählers)
  - inc bzw. dec daher nicht "gleichzeitig" mit copy bzw. delete
  - Unterschied zwischen lokalen und "remote" Referenzen
  - lokales und globales GC (dezentral, echt parallel, typw. hierarchisch)
- Formalisierung des GC-Problems: Operationen C_p, R_p, D_p
- Referenzzähler-Verfahren
  - Problem: "zyklischer Garbage" wird nicht entdeckt
  - bei verteilter Variante: Problem bei decrement *vor* increment



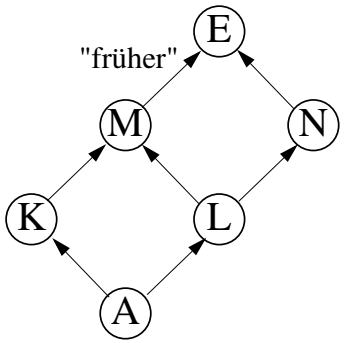
- Lösungen für verteiltes Reference-Counting:
  - prinzipiell: Causal Order garantieren (d.h. indirekte Überholungen vermeiden)
  - "naiv": auf Bestätigung jeder Increment-Nachricht warten
  - ...?

Resümee (22)

- Lösungen für verteiltes Reference-Counting:
  - Varianten von Lermen/Maurer und Rudalics (zwei bis vier Nachrichten pro copy-Operation)
- Weighted Reference Counting (WRC)
  - Kopieren ohne Zusatznachricht: Splitten des Reference Weight
- Analogie (verteiltes) GC <==> verteilte Terminierung
- Transformation GC-Algorithmus --> Algorithmus zur Erkennung der verteilten Terminierung
  - Umformung des Terminierungsproblems in ein GC-Problem
  - darauf gegebenen GC-Algorithmus ansetzen
- Zum Patent der Referenzgewichtsmethode (WRC)
- Patentieren von Algorithmen
- Local Reference Counting (LRC)
  - jede Maschine besitzt für *jedes* Objekt einen (lokalen) Zähler
  - logische Baumstruktur ("Verantwortlichkeit")
  - viele interessante Eigenschaften
  - Migration von Objekten "leicht" zu unterstützen
  - lokal u.U. ein anderes GC-Verfahren (nur global LRC verwenden)
  - IRT / ORT-Tabellen ("Proxy-Objekte"; Bündelung von Referenzen)
  - Aufgabe: Transformation in den Terminierungserkennungsalgorithmus von Dijkstra und Scholten (Literatur nachlesen)

Resümee (23)

- *Verteilte Berechnungen*: Formale Definition (Modellierung!)
  - Partition von Ereignissen, Sende/Empfangsereignisse, Kausalrelation
  - Zeitdiagramme von verteilten Berechnungen; Gummibandtransformation
  - Globale Zustände als Endzustände von Präfixberechnungen (Präfixberechnungen sind linksabgeschlossen bzgl. der Kausalrelation)
  - Menge der Zustände (bzw. Präfixberechnungen) bilden Verband



Berechnung läuft entlang eines "unbestimmten" Weges vom Anfangszustand A zum Endzustand E.

Resümee (24)

- Wellenalgorithmen
  - Information verteilen / einsammeln; Phasen trennen; Ereignisse triggern...
  - Formale Def: ... init < $visit_i$ < conclude ...
  - Visit-Ereignisse bilden einen *Schnitt* (wann senkrechte Schnittlinie möglich?)
  - Bsp.: Echo-Algorithmus, Ring, Stern...
  - min. n-1 Nachrichten, min e Nachrichten bei unbekannten Nachbarn
  - Spannbaum = jeweils erste empfangene Nachricht eines Knotens
  - Halbwellen (Verzicht auf init bzw. conclude); z.B. flooding

- Virtuell gleichzeitiges Markieren mittels flooding
  - Voraussetzung: FIFO-Kanäle

- "Konsistente" Schnittlinien lassen sich senkrecht zeichnen
  - konsistent: keine Nachricht läuft "rückwärts" über die Schnittlinie

- Sequentielle Traversierungsverfahren
  - spezielle Wellenalgorithmen: visit-Ereignisse linear geordnet

- Algorithmus von Tarry (Labyrinth-Problem)
  - Beweisskizze, dass Tarry-Algorithmus ein Traversierungsverfahren ist
  - Depth-First-Search ist Spezialfall des Tarry-Algorithmus

- Traversierungsverfahren von Awerbuch / Cidon

Resümee (25)

- Globale konsistente Schnitte / Zustände

- Schnappschussproblem und -algorithmen

  (1) Färben von Prozessen / Nachrichten; Vermeiden von "Tachyonen"; In-Transit-Nachrichten durch Abgleich von Sende-/Empfangspuffern oder durch Weiterleiten von Kopien an den Initiator

  (2) Chandy/Lamport-Algorithmus: Flooding; FIFO-Kanäle ("flushing"); Problem (?): einige Kanäle sind scheinbar immer leer

Resümee (26)

- Beobachten verteilter Berechnungen
  - Wunsch: lückenlos konsistente Schnappschüsse anzeigen
  - rekonstruiertes Bild des Beobachters
  - ideale und kausaltreue Beobachter
- Kausaltreues Beobachten
  - Beispiele für kausal inkonsistente Beobachtungen
  - Def. kausaltreuer Beobachter
  - Pfade im n-dimensionalen Zustandsgitter ("Hyperwürfel")
- Entdecken globaler Prädikate durch Beobachtung
  - Abhängigkeit von konkreten Beobachtungen ("possible worlds")
  - Wirkung von Handshake- und Barrier-Synchronisation
- Stabile Prädikate

Resümee (26b)

- Schnitte und Vektorzeit
  - Später- / Früher-Relation auf Schnitten
  - Definition konsistenter Schnitte als linksabgeschlossene Ereignismengen
  - Zeitstempel eines Ereignisses als Menge seiner kausalen Vorgänger (Repräsentation durch lokal letztes Ereignis --> Vektorzeit)
- Vektorzeit
  - Interpretation: repräsentiert gesamte kausale Vergangenheit
  - Zeitstempelarithmetik
  - Implementierung (Supremum beim Empfang)
  - Isomorphie der Zeit- und Kausalstruktur
- Anwendung der Vektoruhren
  - kausaltreue Beobachtungen
- Relativistische Struktur der Vektorzeit